This document is published in:
2011 IEEE Congress on Evolutionary Computation (CEC), New Orleans, LA, 5-8 June 2011, pp. 957 - 964.
DOI:10.1109/CEC.2011.5949721

Indicator-based MONEDA: A Comparative Study of Scalability with Respect to Decision Space Dimensions

Luis Martí, Jesús García, Antonio Berlanga and José M. Molina
Group of Applied Artificial Intelligence
Department of Informatics, Universidad Carlos III de Madrid
Av. de la Universidad Carlos III, 22. Colmenarejo 28270 Madrid Spain.
emails: {lmarti,jgherrer}@inf.uc3m.es;{aberlan,molina}@ia.uc3m.es
web: http://www.giaa.inf.uc3m.es.

***Abstract*—The multi-objective neural EDA (MONEDA) was proposed with the aim of overcoming some difficulties of current MOEDAs. MONEDA has been shown to yield relevant results when confronted with complex problems. Furthermore, its performance has been shown to adequately adapt to problems with many objectives. Nevertheless, one key issue remains to be studied: MONEDA scalability with regard to the number of decision variables.**

In this paper has a two-fold purpose. On one hand we propose a modification of MONEDA that incorporates an indicator-based selection mechanism based on the HypE algorithm, while, on the other, we assess the indicator-based MONEDA when solving some complex two-objective problems, in particular problems UF1 to UF7 of the CEC 2009 MOP competition, configured with a progressively-increasing number of decision variables.

I. Introduction

The incorporation of learning as part of the search processes has been nominated as a viable way of improving optimization processes [1]. There are some approaches that perform this task by providing hybrid evolutionary/machine learning method, like, for example, the *learnable evolution model* (LEM) [2]. However, these efforts seem to have been concentrated on single-objective optimization (c. f. [3], [4]).

Another form of carrying out this task to resort to *estimation of distribution algorithms* (EDAs) [5]. This is because of EDAs capacity of learning the problem structure. EDAs replace the application of evolutionary operators with the creation of a statistical model of the fittest elements of the population in a process known as model-building. This model is then sampled to produce new elements.

Nevertheless, the so called *multi-objective EDAs* (MOEDAs) [6] have not live up to their a priori expectations. This can be attributed to the fact that most MOEDAs have limited themselves to transforming single-objective EDAs into a multi-objective formulation by including an existing multi-objective fitness assignment function.

It has been pointed out that the current model-building algorithms of MOEDAs have a set of drawbacks that would prevent those algorithms from yielding substantially better results [7], [8].

In particular, these characteristics are:

- the incorrect treatment of data outliers;
- the loss of population diversity, and;
- the excess of computational effort devoted to finding an optimal model of the fittest elements of the population.

The multi-objective neural EDA (MONEDA) [9], [10] was proposed with the aim of overcoming these difficulties. MONEDA has been shown to yield relevant results when confronted with complex problems. Furthermore, its performance has been shown to adequately adapt to problems with many objectives.

Nevertheless, one key issue remains to be studied: MONEDA scalability with regard to the number of decision variables. This is a rather important issue, as it implies to study MONEDA's ability of exploring high-dimensional search spaces.

One of the consequences of the class of problems previously dealt by MONEDA was the unsuitability of indicator-based selection approaches. As those problems intended to assess MONEDA in the the context of problems with many objectives, the use of performance indicators as part of the selection mechanism was not advised because their high computational requirements.

In this paper has a two-fold purpose. On one hand we propose a modification of MONEDA that incorporates an indicator-based selection mechanism based on the HypE algorithm [11], while, on the other, we assess the indicator-based MONEDA when solving some complex two-objective problems, in particular problems UF1 to UF7 of the CEC 2009 MOP competition [12], configured with a progressively-increasing number of decision variables.

II. Foundations

Many real-world optimization problems involve more than one goal to be optimized. This type of problems is known as *multi-objective optimization problems* (MOPs). A MOP can be expressed as the problem in which a set of *objective functions* $f_1(\boldsymbol{x}), \ldots, f_M(\boldsymbol{x})$ should be jointly optimized;

$$\min \ \boldsymbol{F}(\boldsymbol{x}) = \langle f_1(\boldsymbol{x}), \ldots, f_M(\boldsymbol{x}) \rangle ; \ \boldsymbol{x} \in \mathcal{S} ; \qquad (1)$$

where $\mathcal{S} \subseteq \mathbb{R}^n$ is known as the *feasible set* and could be expressed as a set of restrictions over the decision set, $\mathbb{R}^n$. The image set of $\mathcal{S}$ produced by function vector $\boldsymbol{F}(\cdot)$, $\mathcal{O} \subseteq \mathbb{R}^M$, is called *feasible objective set* or *criterion set*.

The solution to this type of problem is a set of trade-off points. The optimality of a solution can be expressed in terms of the Pareto dominance relation.

Definition 1 (Pareto dominance relation): In the optimization problem (1) and having $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{S}$, $\boldsymbol{x}$ is said to dominate $\boldsymbol{y}$ (expressed as $\boldsymbol{x} \prec \boldsymbol{y}$) iff $\forall f_j$, $f_j(\boldsymbol{x}) \leq f_j(\boldsymbol{y})$ and $\exists f_i$ such that $f_i(\boldsymbol{x}) < f_i(\boldsymbol{y})$.

Definition 2 (non-dominated subset): In problem (1) and having the set $\mathcal{A} \subseteq \mathcal{S}$. $\mathcal{A}^*$, the *non-dominated subset* of $\mathcal{A}$, is defined as

$$\mathcal{A}^* = \{\boldsymbol{x} \in \mathcal{A} \,|\, \nexists \boldsymbol{x}' \in \mathcal{A} : \boldsymbol{x}' \prec \boldsymbol{x}\}.$$

The solution of (1) is $\mathcal{S}^*$, the non-dominated subset of $\mathcal{S}$. $\mathcal{S}^*$ is known as the *efficient set* or *Pareto-optimal set* [13]. Its image in objective set is known as the *Pareto-optimal front*, $\mathcal{O}^*$.

If problem (1) has certain characteristics, e. g., linearity or convexity of the objective functions or convexity of $\mathcal{S}$, the efficient set can be determined by mathematical programming approaches [13]. However, in the general case, finding the solution of (1) is an *NP*–complete problem [**?**]. In this case, heuristic or metaheuristic methods can be applied in order to have solutions of practical value at an admissible computational cost.

A broad range of heuristic and metaheuristic approaches has been used to address MOPs [13]. Of these, multi-objective evolutionary algorithms (MOEAs) [14], [15] have been found to be a competent approach in a wide variety of application domains.

As finding the explicit formulation of $\mathcal{S}^*$ is often impossible, generally, an algorithm solving (1) yields a discrete non-dominated set, $\mathcal{P}^*$, that approximates $\mathcal{S}^*$. The image of $\mathcal{P}^*$ in objective set, $\mathcal{PF}^*$, is known as the *non-dominated front*.

Determining how good a given $\mathcal{P}^*$, or $\mathcal{PF}^*$, is not only a key but also a particularly complex task. It necessarily implies a reduction from an M–dimensional set to a scalar value. Therefore, as in any dimensionality reduction, valuable information could be lost, leading to invalid conclusions. This point has been well documented [16].

Nevertheless, there are some community-accepted indicators of the quality of a solution $\mathcal{P}^*$ [17]. Such indicators can be grouped into three broad categories:

1) distance from the elements of $\mathcal{PF}^*$ to their corresponding closest element of $\mathcal{O}^*$, which measures how close the solution is to the optima;
2) distance from every element of $\mathcal{O}^*$ to its closest element of $\mathcal{PF}^*$, which complements the first class of indicators and expresses how well $\mathcal{PF}^*$ covers $\mathcal{O}^*$, and
3) distribution of the elements of $\mathcal{P}^*$ and $\mathcal{PF}^*$, which gauges how well spread the elements of these sets are.

One of the interesting side-effects of the appearance of performance indicators is that their use has been extended to the population ranking and selection mechanisms of MOEAs. For this purpose the hypervolume indicator has been particularly studied.

The hypervolume indicator, $I_{\text{hyp}}(\mathcal{A})$, [17] computes the volume of the region, H, delimited by a given set of points, $\mathcal{A}$, and a set of reference points, $\mathcal{N}$.

$$I_{\text{hyp}}(\mathcal{A}) = \text{volume}\left(\bigcup_{\forall \boldsymbol{a} \in \mathcal{A}; \forall \boldsymbol{n} \in \mathcal{N}} \text{hypercube(a, n)}\right). \quad (2)$$

Therefore, larger values of the indicator will correspond to better solutions.

To measure the absolute performance of an algorithm the reference points should ideally be nadir points. These points are the worst elements of $\mathcal{O}$, or, in other words, the elements of $\mathcal{O}$ that do not dominate any other element. To contrast the relative performance of two sets of solutions, though, one can be used as the reference set. These matters are further detailed in [16], [17].

Having $\mathcal{N}$, the computation of the indicator is a non-trivial problem. Indeed, its determination is known to be computationally intensive, thus rendering it is unsuitable for problems with many objectives.

A lot of research have focused on improving the computational complexity of this indicator (c. f. [18]). According to the most recent results, the indicator is currently known to be $O(n \log n + n^{M/2})$ [18] for more than three objectives ($M > 3$); $O(n \log n)$ for $M = 2, 3$ [19].

As the main drawback of this indicator is its computational requirements and poor scalability some approaches has been proposed to estimate its value instead of explicitly computing it. Among these approaches we find the Monte Carlo estimation put forward as part of the HypE algorithm [11].

III. Estimation of Distribution Algorithms

An evolutionary algorithm can be characterized by how it implements a set of processes, in particular,

- *Mating selection*: that determines the degree at which individuals in the population will take part in the generation of new (offspring) individuals.
- *Variation*: which applies a range of evolutionary operators to synthesize offspring individuals from the current (parent) population. This process is supposed to prime the most fit individuals so they play a bigger role in the generation of the offspring.
- *Environmental selection*: that merges the parent and offspring individuals to produce the population that will be used in the next iteration. This process often involves the deletion of some individuals using a given criterion in order to keep the number of individuals bellow a certain threshold.

Estimation of distribution algorithms (EDAs) [5] have been hailed as a landmark in the progress of evolutionary algorithms. They replace the application of evolutionary operators in the variation process with the creation of a statistical model of the fittest elements of the population using a machine

learning algorithm. This *model–building algorithm* is the key feature that differentiates EDAs from other evolutionary approaches. The constructed model is then sampled to produce new elements.

The extension of EDAs to the multi-objective domain has lead to multi-objective optimization EDAs (MOEDAs) [6]. Most MOEDAs have limited themselves to port single objective EDAs to the multi-objective domain by incorporating some features taken from MOEAs, mainly the selection strategies.

IV. Indicator-based Multi-objective Neural EDA

The indicator-based multi-objective neural EDA (MONEDA/I) is a MOEDA that uses a modified growing neural gas (MB–GNG) network as its model–building algorithm and the selection processes proposed by the Hypervolume Estimation Algorithm for Multiobjective Optimization (HypE) algorithm. The MB–GNG network is a custom–made model–building algorithm devised to cope with the specifications of the task and that will be described later on. Similarly, the HypE algorithm is one of the most effective approaches to MOPs. Therefore, it could be hypothesised that the resulting hybrid algorithm would exhibit the outstanding properties of its constituents.

A. HypE Selection

For population ranking and mating selection HypE determines the hypervolume for each population element. Depending on the number of objective HypE calculates or estimates via a Monte Carlo sampling the hypervolume. As to mating selection, binary tournament selection is used.

Environmental selection primes the most promising solutions from the union of parent (current) population and offspring; more precisely, it creates a new population by carrying out the following two steps

- The union of parents and offspring is divided into disjoint non-dominated partitions. Starting with the lowest dominance depth level, the partitions are moved one by one to the new population until the first partition is reached that cannot be transferred completely.
- The partition that only fits partially is then processed by calculating the fitness values of the individuals and the individual with the worst fitness is removed. This procedure is repeated until the partition has been reduced to the desired size.

Further details of HypE can be obtained from it corresponding paper.

B. Model–Building with Growing Neural Gas

Clustering algorithms have been used as part of the model–building algorithms of EDAs and MOEDAs. As a foundation for our proposal we have chosen the growing neural gas (GNG) network [20]. GNG networks are intrinsic self–organizing neural networks based on the neural gas [21] model. This model relies in a competitive Hebbian learning rule [22]. It creates an ordered topology of inputs classes and associates a cumulative error to each. The topology and the cumulative errors are conjointly used to determine how new classes should be inserted.

The model–building GNG (MB–GNG) [8] is an extension of the original (unsupervised) GNG. MB–GNG creates a quantization of the input space using a modified version of the GNG algorithm and then computes the deviations associated to each node.

We have added a cluster repulsion mechanism [23] to the original GNG formulation. This enhancement fosters exploration of the input space as it makes each cluster to represent a distinctive zone of the space.

MB–GNG is a one layer network that defines each class as a local Gaussian density and adapts them using a local learning rule. The layer contains a set of nodes $\mathcal{C} = \{c_1, \ldots, c_{N^*}\}$, with $N_0 \leq N^* \leq N_{\max}$. Here N_0 and $N_{\max}$ represent the initial and maximal amount of nodes in the network.

A node c_i consists of a center, $\boldsymbol{\mu}_i$, deviations, $\boldsymbol{\sigma}_i$, an accumulated error, ξ_i, and a set of edges that define the set of topological neighbors of c_i, $\mathcal{V}_i$. Each edge has an associated age, $\nu_{i,j}$.

The dynamics of a GNG network consists of three concurrent processes: network adaptation, node insertion and node deletion. The combined use of these three processes renders GNG training Hebbian in spirit [22].

The network is initialized with N_0 nodes with their centers set to randomly chosen inputs. A training iteration starts after an input $\boldsymbol{x}$ is randomly selected from the training data set, Ψ. Then two nodes are selected for being the closest ones to $\boldsymbol{x}$. The *best–matching node*, c_b,

$$b = \underset{i=1,\ldots,N^*}{\arg\min}\, d\left(\boldsymbol{\mu}_i, \boldsymbol{x}\right), \tag{3}$$

is the closest node to $\boldsymbol{x}$. Consequently, the *second best–matching node*, $c_{b'}$, is determined as

$$b' = \underset{i=1,\ldots,N^*; i\neq b}{\arg\min}\, d\left(\boldsymbol{\mu}_i, \boldsymbol{x}\right). \tag{4}$$

Here $d\left(\boldsymbol{a}, \boldsymbol{b}\right)$ is a distance metric. For this work we have used $d(\cdot)$ defined as

$$d\left(\boldsymbol{a}, \boldsymbol{b}\right) = \left\|\boldsymbol{a} - \boldsymbol{b}\right\|. \tag{5}$$

If $c_{b'}$ is not a neighbor of c_b then a new edge is established between them $\mathcal{V}_b = \mathcal{V}_b \cup \{c_{b'}\}$ with zero age, $\nu_{b,b'} = 0$. If, on the other case, $c_{b'} \in \mathcal{V}_b$ the age of the corresponding edge is reset $\nu_{b,b'} = 0$.

At this point, the age of all edges is incremented in one. If an edge is older than the maximum age, $\nu_{i,j} > \nu_{\max}$, then the edge is removed. If a node becomes isolated from the rest it is also deleted.

Clustering error is then added to the best–matching node error accumulator,

$$\Delta\xi_b = d\left(\boldsymbol{\mu}_i, \boldsymbol{x}\right)^2. \tag{6}$$

After that, learning takes place in the best–matching node and its neighbors with rates ϵ_{best} and ϵ_{vic}, respectively. For c_b

adaptation follows the rule originally used by GNG

$$\Delta\boldsymbol{\mu}_b = \epsilon_{\text{best}} \left(\boldsymbol{x} - \boldsymbol{\mu}_b\right) . \tag{7}$$

However for c_b's neighbors a cluster repulsion term [23] is added to the original formulation, that is, $\forall c_v \in \mathcal{V}_b$,

$$\begin{aligned}\Delta\boldsymbol{\mu}_v = \quad & \epsilon_{\text{vic}} \left(\boldsymbol{x} - \boldsymbol{\mu}_v\right) \\ & +\beta e^{\left(-\frac{d(\boldsymbol{\mu}_v,\boldsymbol{\mu}_b)}{\zeta}\right)} \frac{\sum_{c_u \in \mathcal{V}_b} d(\boldsymbol{\mu}_u,\boldsymbol{\mu}_b)}{|\mathcal{V}_b|} \frac{(\boldsymbol{\mu}_v - \boldsymbol{\mu}_b)}{d(\boldsymbol{\mu}_v,\boldsymbol{\mu}_b)} .\end{aligned} \tag{8}$$

Here β is an integral multiplier that defines the amplitude of the repulsive force while ζ controls the weakening rate of the repulsive force with respect to the distance between the nodes' centers. This approach was already used as part of the robust GNG [24] and it has proved itself useful for obtaining a good spread of the clusters in the inputs' space. In the aforementioned work its stated that the adaptation rule is not sensitive with respect to its parameters. We have set them to $\beta = 2$ and $\zeta = 0.1$ as suggested in [24].

If the current iteration is an integer multiple of T_+ and N^* is smaller than $N_{\max}$ then a new neuron is inserted to the network. First, the node with largest error, c_e, is selected the node. Then the worst node among its neighbors, $c_{e'}$, is located. Then N^* is incremented and the new node, c_{N^*}, is inserted between the two nodes,

$$\boldsymbol{\mu}_{N^*} = 0.5 \left(\boldsymbol{\mu}_e + \boldsymbol{\mu}_{e'}\right) . \tag{9}$$

The edge between c_e and $c_{e'}$ is removed and two new edges connecting c_{N^*} with c_e and $c_{e'}$ are created. The accumulated errors are reduced in a rate $0 \leq \delta_I \leq 1$ by letting

$$\xi_e = \delta_I \xi_e , \ \xi_{e'} = \delta_I \xi_{e'} . \tag{10}$$

The error of the newly created node is computed as

$$\xi_{N^*} = 0.5(\xi_e + \xi_{e'}) . \tag{11}$$

Finally, the errors of all nodes are decreased by a factor δ_{G},

$$\xi_i = \delta_{\text{G}} \xi_i, \ i = 1, \ldots, N^* . \tag{12}$$

Stopping the learning of GNG is a non–trivial issue shared by the rest of clustering algorithms and all reiterative heuristic algorithms. As we are interested to cover the inputs space as much as possible we will stop if, after a learning epoch, the standard deviation of the accumulated errors is smaller than a certain threshold, ρ,

$$\sqrt{\frac{1}{N^*} \sum_{i=1}^{N^*} (\xi_i - \overline{\xi})^2} < \rho . \tag{13}$$

This means that we will stop when the outliers are as better represented as possible.

After training has ended the deviations, $\boldsymbol{\sigma}_i$, of the nodes must be computed. For this task we employ the unbiased estimator of the deviations detail in the following algorithm

Set $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{N^*} = \mathbf{0}$ and $n_1, \ldots, n_{N^*} = 0$.
for all $\boldsymbol{x} \in \Psi$ **do**
Determine the closest node, c_c to $\boldsymbol{x}$.
$\boldsymbol{s}_c = \boldsymbol{s}_c + (\boldsymbol{x} - \boldsymbol{\mu}_c)^2$.
$n_c = n_c + 1$.
Compute the deviations as $\boldsymbol{\delta}_i = \sqrt{\frac{\boldsymbol{s}_i}{n_i}}$.

MB–GNG parameters: N_0, $\nu_{\max}$, ϵ_{b}, ϵ_{v}, δ_{I}, δ_{G} and ρ.
MONEDA parameters: n_{pop}, α, γ and ω.
Let $t = 0$.
Randomly generate the initial population P_0 with z individuals.
repeat
Determine $\hat{\mathcal{P}}_t$ using HypE mating selection.
Train MB–GNG network with $\hat{\mathcal{P}}_t$ training data set and $N_{\max} = \left\lfloor \gamma \left| \hat{\mathcal{P}}_t \right| \right\rfloor$.
Sample $\lfloor \omega \left| \mathcal{P}_t \right| \rfloor$ from the MB–GNG.
Produce $\mathcal{P}_{t+1}$ by applying HypE environmental selection procedure.
$t = t + 1$.
until end condition is met
Determine the set of non-dominated individuals of $\mathcal{P}_t$, $\mathcal{P}_t^*$.
return $\mathcal{P}_t^*$ as the algorithm's solution.

Fig. 1. Algorithmic representation of MONEDA/I.

C. MONEDA/I Algorithm

MONEDA maintains a population of individuals, $\mathcal{P}_t$, with t as the current iteration. The algorithm's workflow is similar to other EDAs. It starts from a random initial population $\mathcal{P}_0$ of n_{pop} individuals. It then proceeds to sort the individuals using the HypE mating selection process.

A set $\hat{\mathcal{P}}_t$ containing $\lfloor \alpha \left| \mathcal{P}_t \right| \rfloor$ selected elements is extracted from the sorted version of $\mathcal{P}_t$,

$$\left| \hat{\mathcal{P}}_t \right| = \lfloor \alpha \left| \mathcal{P}_t \right| \rfloor . \tag{14}$$

A MB–GNG network is then trained using $\hat{\mathcal{P}}_t$ as training data set. In order to have a controlled relation between size of $\hat{\mathcal{P}}_t$ and the maximum size of the network, $N_{\max}$, these two sizes are bound by the rate $\gamma \in (0, 1]$,

$$N_{\max} = \left\lceil \gamma \left| \hat{\mathcal{P}}_t \right| \right\rceil . \tag{15}$$

The resulting Gaussian kernels are sampled to produce an amount $\lfloor \omega \left| \mathcal{P}_t \right| \rfloor$ of new individuals. This offspring is combined with the parent population as previously described for the HypE environmental selection. The set obtained is then united with best elements, $\hat{\mathcal{P}}_t$, to form the population of the next iteration $\mathcal{P}_t$.

Iterations are repeated until a given stopping criterion is met. The output of the algorithm is the set of non–dominated solutions of $\mathcal{P}_t$, $\mathcal{P}_t^*$. The outline of MONEDA is presented in algorithmic form on Fig. 1.

V. Experiments

Experiments are required in order to understand the effectiveness or our proposal. As already commented, we will be focusing of a set of problems previously proposed in the CEC 2009 MOP competition [12]. From the set of problems proposed there we selected the unconstrained optimization problems UF1 to UF7. These are two-objective problems that can be configured to have any desired number of variables. These problems are well-known for the complexity of their Pareto-optimal sets and fronts. They were selected in order to be able to plot the results and to visually compare results.

The problems were configured with an escalating complexity by configure them with 20, 30, 40, 50 and 60 variables. Experiments were carried out under the PISA framework [25]. Each experiment configuration was repeated 30 times in order to have statistically valid judgements. Algorithms runs were controlled by MGBM criterion [26].

The best results of MONEDA/I for each problem and dimension can be examined in Figs. 2 and 3. It is noticeable that in some cases MONEDA/I yielded better results when facing to lower dimensional problems, like in the case of UF1, UF2, UF3 and UF7. In some other cases, like, in particular UF4, larger numbers of variables prompted better results. This result is worth of notice when compared with those obtained by the participants of the aforementioned competition. By simple visual inspection we can assert that our approach yield better results. It must be pointed out, however, that thanks to our different stopping condition MONEDA/I, and the other algorithms involved in the experiments later on, were left to run for a longer period of time.

Although many interesting reflections can emerge from examining these plots it is clear than a more sound interpretation of results is required. In particular, it is needed to compare MONEDA/I with similar approaches. That is why we included in the experiments the original Pareto-based MONEDA, two well-known MOEDAs, in particular, naïve MIDEA [27] and MrBOA [28], two state-of-the-art MOEAs, that is, HypE [11] and MOEA/D [29] and the "classical" NSGA-II [30] MOEA.

The quality of the solutions is determined by the use of the hypervolume indicator [31]. The statistical validity of the judgment of the results calls for the application of statistical hypothesis tests. It has been previously remarked by different authors that the Mann-Whitney-Wilcoxon U test [32] is particularly suited for experiments in the context of multi-objective evolutionary optimization [31]. This test is commonly used as a non-parametric method for testing equality of population medians. In our case we performed pair-wise tests on the significance of the difference of the indicator values yielded by the executions of the algorithms. A significance level, α, of 0.05 was used for all tests.

The visual analysis of the results is rather difficult as it implies cross-examining and comparing the results presented separately. That is why we decided to adopt a more integrative representation such as the one proposed in [33]. That is, for a given set of algorithms $A_1,\ldots, A_K$, a set of P test problem instances $\Phi_{1,m},\ldots,\Phi_{P,m}$, configured with m objectives, the function $\delta(\cdot)$ is defined as

$$\delta\left(A_i, A_j, \Phi_{p,m}\right) = \begin{cases} 1 & \text{if } A_i \gg A_j \text{ solving } \Phi_{p,m} \\ 0 & \text{in other case} \end{cases}, \tag{16}$$

where the relation $A_i \gg A_j$ defines if A_i is significantly better than A_j when solving the problem instance $\Phi_{p,m}$, as computed by the statistical tests previously described.

Relying on $\delta(\cdot)$, the performance index $P_{p,m}(A_i)$ of a given algorithm A_i when solving $\Phi_{p,m}$ is then computed as

$$P_{p,m}\left(A_i\right) = \sum_{j=1;j\neq i}^{K} \delta\left(A_i, A_j, \Phi_{p,m}\right). \tag{17}$$

This index intends to summarize the performance of each algorithm with regard to its peers.

Figs. 4 and 5 exhibits the results computing the performance indexes grouped by problems and dimensions.

Fig. 4 represents the mean performance indexes yielded by each algorithm when solving each problem in all of its configured objective dimensions,

$$\bar{P_p}\left(A_i\right) = \frac{1}{|\mathcal{M}|} \sum_{m\in\mathcal{M}} P_{p,m}\left(A_i\right). \tag{18}$$

It is worth noticing that MONEDA/I has better overall results with respect to the other algorithms in all problems. As it could be expected, the use of indicator-based selection in MONEDA has yielded better results than the original MONEDA. Indicator-based MONEDA and the indicator-based MOEAs have a similar performance. It can be hypothesized that these results can be biased by the three objective problems, having dramatic differences in their results with respect to the rest of the dimensions considered.

This situation is clarified in Fig. 5, which presents the mean values of the index computed for each dimension

$$\bar{P_m}\left(A_i\right) = \frac{1}{P} \sum_{p=1}^{P} P_{p,m}\left(A_i\right). \tag{19}$$

These experiments prompt some interesting reflections. First, it is noticeable that the results of MONEDA/I are substantially better in many cases. Furthermore, the combination of MONEDA with HypE is capable of reaching better results than MONEDA and HypE separately. Similarly, the results obtained are comparable and in many cases better than those of MOEA/D, which is one of the current best performing MOEAS. It is also noticeable that those algorithms that employ an indicator-based selection produce better results than those that are Pareto-based, a result that should be taken into account in subsequent studies.

VI. Conclusion

In this paper we have proposed a new version of the MONEDA algorithm that incorporates an indicator-based selection taken from the HypE algorithm. Thanks to this modification the new algorithm is capable of outperforming it constituents.

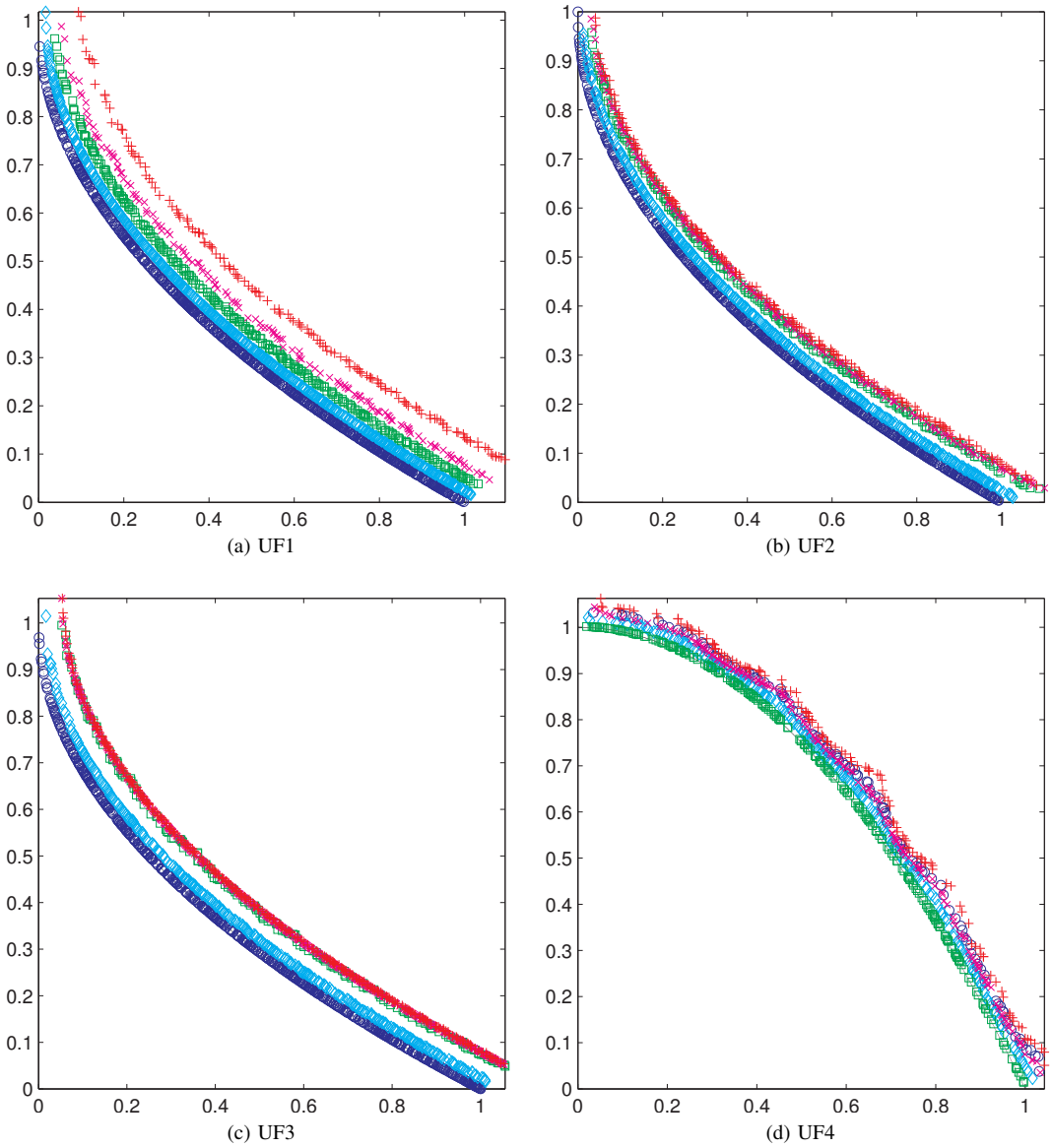


Fig. 2. Non-dominated fronts obtained by the best run of MONEDA/I when solving problems UF1, UF2, UF3 and UF4. The results shown correspond to different numbers of decision variables. In particular, 20 (○), 30 (◇), 40 (□), 50 (×) and 60 (+) variables.

Therefore, a new line of MOEDA development comes to light: indicator-based MOEDAs. Perhaps, the use of this relatively novel form of carrying out the selection processes would ease the diversity loss issue that is undermining current MOEDAs.

Although the experiments clearly show the advantage of employing MONEDA/I, further experimentation is obviously required.

ACKNOWLEDGMENT

This work was supported by projects CICYT TIN2008-06742-C02-02/TSI, CICYT TEC2008-06732-C02-02/TEC, SINPROB, CAM CONTEXTS S2009/TIC-1485 and DPS2008-07029-C02-02.

REFERENCES

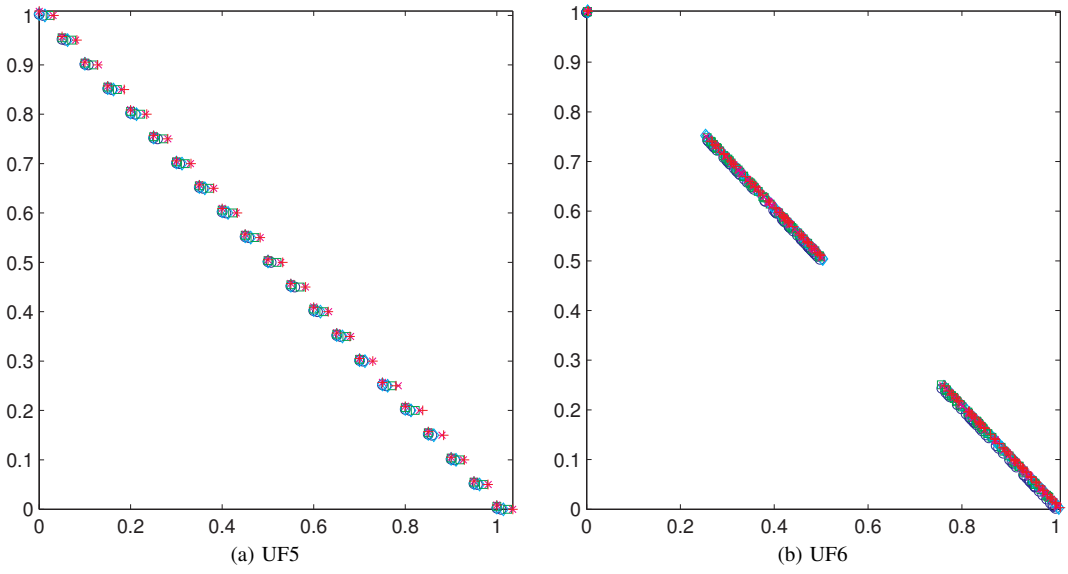
[1] D. W. Corne, "Single objective = past, multiobjective = present, ??? = future," in *2008 IEEE Conference on Evolutionary Computation (CEC), part of 2008 IEEE World Congress on Computational Intelligence (WCCI 2008)*, Z. Michalewicz, Ed. Piscataway, New Jersey: IEEE Press, 2008.

[2] R. S. Michalski, "Learnable evolution model: Evolutionary processes guided by machine learning," *Machine Learning*, vol. 38, pp. 9–40, 2000.

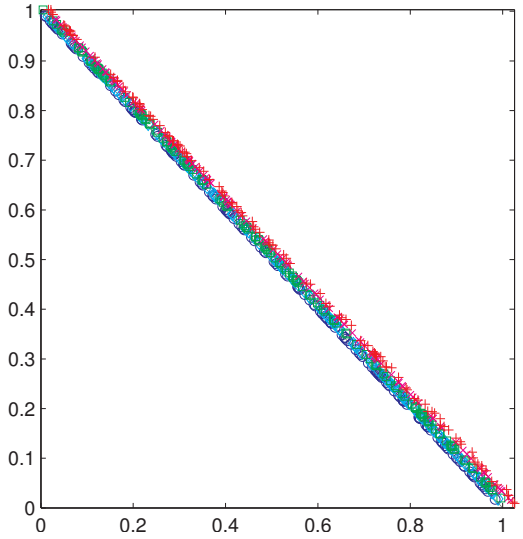
[3] G. Sheri and D. W. Corne, "The simplest evolution/learning hybrid: LEM with KNN," in *IEEE World Congress on Computational Intelligence*. Hong Kong, China: IEEE Press, 2008, pp. 3244–3251.

[4] ——, "Learning-assisted evolutionary search for scalable function optimization: LEM(ID3)," in *IEEE World Congress on Computational Intelligence*. Barcelona, Spain: IEEE Press, 2010.

[5] J. A. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea, Eds., *Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms*. Springer–Verlag, 2006.



(a) UF5 (b) UF6



(c) UF7

Fig. 3. Non-dominated fronts obtained by the best run of MONEDA/I when solving problems UF5, UF6 and UF7. The results shown correspond to different numbers of decision variables. In particular, 20 (○), 30 (◇), 40 (□), 50 (×) and 60 (+) variables.

[6] M. Pelikan, K. Sastry, and D. E. Goldberg, "Multiobjective estimation of distribution algorithms," in *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*, ser. Studies in Computational Intelligence, M. Pelikan, K. Sastry, and E. Cantú-Paz, Eds. Springer–Verlag, 2006, pp. 223–248.

[7] L. Martí, J. García, A. Berlanga, C. A. Coello Coello, and J. M. Molina, "On current model-building methods for multi-objective estimation of distribution algorithms: Shortcommings and directions for improvement," Grupo de Inteligencia Artificial Aplicada, Universidad Carlos III de Madrid, Colmenarejo, Spain, Tech. Rep. GIAA2010E001, 2010. [Online]. Available: http://www.giaa.inf.uc3m.es/miembros/lmarti/model-building

[8] ——, "MB-GNG: Addressing drawbacks in multi-objective optimization estimation of distribution algorithms," *Operations Research Letters*, vol. 39, no. 2, pp. 150–154, 2011.

[9] L. Martí, J. García, A. Berlanga, and J. M. Molina, "MONEDA: Scalable multi–objective optimization with a neural network–based estimation of distribution algorithm," Grupo de Inteligencia Artificial Aplicada, Universidad Carlos III de Madrid, Colmenarejo, Spain, Tech. Rep. GIAA2010E002, 2010. [Online]. Available: http://www.giaa.inf.uc3m.es/miembros/lmarti/moneda

[10] ——, "Introducing MONEDA: Scalable multiobjective optimization with a neural estimation of distribution algorithm," in *GECCO '08: 10th Annual Conference on Genetic and Evolutionary Computation*, M. Keizer, G. Antoniol, C. Congdon, K. Deb, B. Doerr, N. Hansen, J. Holmes, G. Hornby, D. Howard, J. Kennedy, S. Kumar, F. Lobo, J. Miller, J. Moore, F. Neumann, M. Pelikan, J. Pollack, K. Sastry, K. Stanley, A. Stoica, E. G. Talbi, and I. Wegener, Eds. New York, NY, USA: ACM Press, 2008, pp. 689–696, eMO Track "Best Paper" Nominee.

[11] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," *Evolutionary Computation*, vol. 19, no. 1, pp. 45–76, 2011. [Online]. Available: http://www.mitpressjournals.org/doi/abs/10.1162/EVCO_a_00009

[12] Q. Zhang, A. Zhou, S. Zhao, P. Suganthan, W. Liu, and S. Tiwari, "Multiobjective optimization test instances for the CEC 2009 special session and competition," University of Essex, Colchester, UK and Nanyang Technological University, Singapore, Tech. Rep., 2009.

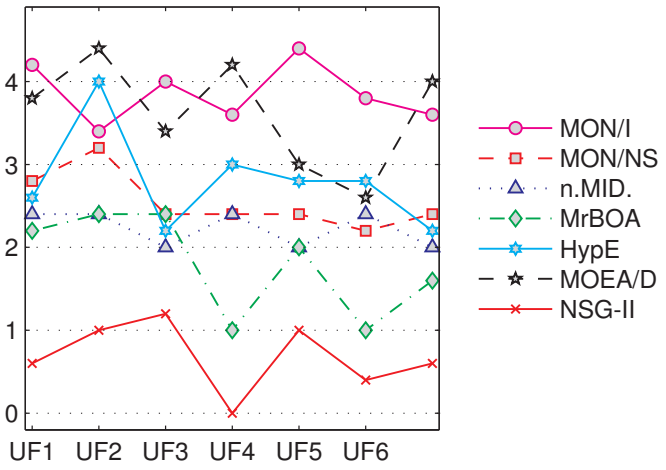


Fig. 4. Mean values of the performance index indicator-based MONEDA (MON/I), MONEDA with NSGA-II selection (MON/NS), naïve MIDEA (n.MID), MrBOA, HypE, MOEA/D and NSGA-II (NSG-II) across the different problems, $\bar{P}_p()$.

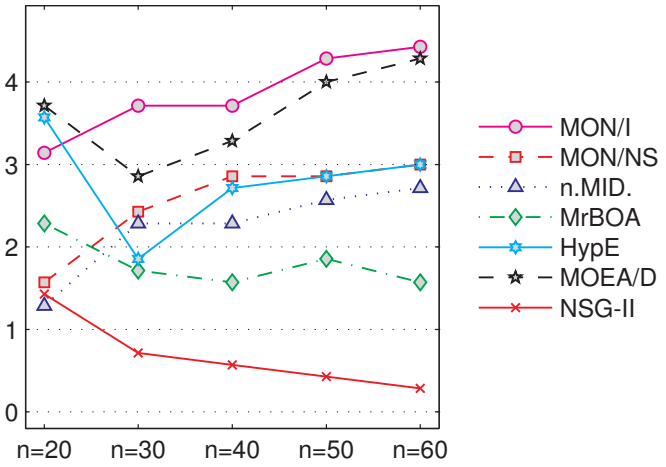


Fig. 5. Mean values of the performance index across the different space dimensions, $\bar{P}_m$. See Fig. 4 for a description of the acronyms.

[13] J. Branke, K. Miettinen, K. Deb, and R. Słowiński, Eds., *Multiobjective Optimization*, ser. Lecture Notes in Computer Science. Berlin/Heidelberg: Springer–Verlag, 2008, vol. 5252.

[14] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd ed., ser. Genetic and Evolutionary Computation. New York: Springer, 2007. [Online]. Available: http://www.springer.com/west/home/computer/foundations?SGWID=4-156-22-173660344-0

[15] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. Chichester, UK: John Wiley & Sons, 2001, ISBN 0-471-87339-X.

[16] E. Zitzler, M. Laumanns, L. Thiele, C. M. Fonseca, and V. Grunert da Fonseca, "Why Quality Assessment of Multiobjective Optimizers Is Difficult," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2002)*, W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. Potter, A. Schultz, J. Miller, E. Burke, and N. Jonoska, Eds. San Francisco, California: Morgan Kaufmann Publishers, July 2002, pp. 666–673.

[17] J. Knowles, L. Thiele, and E. Zitzler, "A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers," Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland, 214, Feb. 2006, revised version.

[18] N. Beume, "S–metric calculation by considering dominated hypervolume as Klee's measure problem," *Evolutionary Computation*, vol. 17, no. 4, pp. 477–492, 2009, pMID: 19916778. [Online]. Available: http://www.mitpressjournals.org/doi/abs/10.1162/evco.2009.17.4.17402

[19] C. M. Fonseca, L. Paquete, and M. López-Ibánez, "An improved dimension–sweep algorithm for the hypervolume indicator," in *2006 IEEE Congress on Evolutionary Computation (CEC'2006)*, 2006, pp. 1157–1163.

[20] B. Fritzke, "A growing neural gas network learns topologies," in *Advances in Neural Information Processing Systems*, G. Tesauro, D. S. Touretzky, and T. K. Leen, Eds. Cambridge, MA: MIT Press, 1995, vol. 7, pp. 625–632.

[21] T. M. Martinetz, S. G. Berkovich, and K. J. Shulten, "Neural–gas network for vector quantization and its application to time–series prediction," *IEEE Transactions on Neural Networks*, vol. 4, pp. 558–560, 1993.

[22] T. M. Martinetz, "Competitive Hebbian learning rule forms perfectly topology preserving maps," in *International Conference on Artificial Neural Networks (ICANN'93)*. Amsterdam: Springer–Verlag, 1993, pp. 427–434.

[23] H. Timm, C. Borgelt, C. Doring, and R. Kruse, "An extension to possibilistic fuzzy cluster analysis," *Fuzzy Sets and Systems*, vol. 147, no. 1, pp. 3–16, October 2004.

[24] A. K. Qin and P. N. Suganthan, "Robust growing neural gas algorithm with application in cluster analysis," *Neural Networks*, vol. 17, no. 8–9, pp. 1135–1148, 2004.

[25] S. Bleuler, M. Laumanns, L. Thiele, and E. Zitzler, "PISA—A Platform and Programming Language Independent Interface for Search Algorithms," in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele, Eds. Faro, Portugal: Springer. Lecture Notes in Computer Science. Volume 2632, April 2003, pp. 494–508.

[26] L. Martí, J. García, A. Berlanga, and J. M. Molina, "An approach to stopping criteria for multi–objective optimization evolutionary algorithms: The MGBM criterion," in *2009 IEEE Conference on Evolutionary Computation (CEC 2009)*. Piscataway, New Jersey: IEEE Press, 2009, pp. 1263–1270.

[27] P. A. N. Bosman and D. Thierens, "The naïve MIDEA: A baseline multi–objective EA," in *Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005*, C. A. Coello Coello, A. Hernández Aguirre, and E. Zitzler, Eds. Guanajuato, México: Springer. Lecture Notes in Computer Science Vol. 3410, March 2005, pp. 428–442.

[28] C. W. Ahn, *Advances in Evolutionary Algorithms. Theory, Design and Practice*. Springer, 2006, ISBN 3-540-31758-9.

[29] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[30] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, April 2002.

[31] J. Knowles, L. Thiele, and E. Zitzler, "A tutorial on the performance assessment of stochastic multiobjective optimizers," Computer Engineering and Networks Laboratory (TIK), ETH Zurich, TIK Report 214, 2006.

[32] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *Annals of Mathematical Statistics*, vol. 18, pp. 50–60, 1947.

[33] J. Bader, "Hypervolume-Based Search for Multiobjective Optimization: Theory and Methods," Ph.D. dissertation, ETH Zurich, Switzerland, 2010.